

CLAIMS

What is claimed is:

1. A method for generating factored assert chains from assert statements in a program using a compiler, the program having a dominator tree, wherein the method has as input parameter  $X$  initially assigned an entry node of said dominator tree, the method comprising:

for each statement  $S$  in  $X$ :

finding each variable use in the statement;

for each variable use in the statement:

determining if there is an available assert statement which defines information about the corresponding variable for the variable use;

making an assert chain from the variable use to the available assert statement if there is an available assert statement which defines information about the corresponding variable for the variable use; and

iteratively calling the method using each child of  $X$  in the dominator tree as a parameter.

2. The method of claim 1, further comprising saving a current value in a map of assert statements for each variable.

3. The method of claim 1, further comprising:

determining if the statement is an assert statement after said making;

adding a mapping from each variable use in the statement to an assert statement if the statement is an assert statement; and

deleting a mapping of any reference to an assert statement for each definition in the statement if the statement is not an assert statement.

4. The method of claim 1, further comprising restoring a current value of a map of assert statements for each variable after said iteratively calling.

5. The method of claim 1, wherein an assert statement is a statement identifying known information regarding a variable.

6. A method for generating factored assert chains for a program in a compiler, the method comprising:

generating one or more assert statements in a basic block of the program;  
creating a dominator tree for said basic block;  
creating an initialized map of assert statements for each variable in said basic block; and  
calling an assert chain search procedure with entry as a parameter, wherein entry is a root node of said dominator tree.

7. The method of claim 6, wherein said generating includes:

finding an if...then...else statement in said basic block, said if...then...else statement having a condition, a then portion, and an else portion;  
inserting an assert statement indicating that said condition is true in said then portion; and  
inserting an assert statement indicating that said condition is false in said else portion.

8. The method of claim 6, wherein said generating includes:  
finding a do loop, said do loop having a non-constant stride; and  
inserting an assert statement indicating that said stride is not equal to zero.
9. The method of claim 6, wherein said assert chain search procedure has as input parameter *X*, the assert chain search procedure comprising:  
for each statement *S* in *X*:  
    finding each variable use in the statement;  
    for each variable use in the statement:  
        determining if there is an available assert statement which defines  
information about the corresponding variable for the variable use;  
        making an assert chain from the variable use to the available assert  
statement if there is an available assert statement which defines information about  
the corresponding variable for the variable use; and  
iteratively calling the method using each child of *X* in the dominator tree as a parameter.
10. The method of claim 9, wherein said assert chain search procedure further comprises  
saving a current value in a map of assert statements for each variable.
11. The method of claim 9, wherein said assert chain search procedure further comprises:  
determining if the statement is an assert statement after said making;  
adding a mapping from each variable use in the statement to an assert statement if the  
statement is an assert statement; and

deleting a mapping of any reference to an assert statement for each definition in the statement if the statement is not an assert statement.

12. The method of claim 9, wherein said assert chain search procedure further comprises restoring a current value of a map of assert statements for each variable after said iteratively calling.
13. The method of claim 9, wherein an assert statement is a statement identifying known information regarding a variable.
14. An apparatus for generating factored assert chains from assert statements, the apparatus comprising:
  - a statement traverser;
  - a variable use finder coupled to said statement traverser;
  - a variable use traverser coupled to said statement traverser and said variable use finder;
  - an available assert statement determiner coupled to said variable user traverser;
  - an assert chain creator coupled to said available assert statement determiner and to said variable use traverser; and
  - an assert chain search procedure iterative caller coupled to said statement traverser.
15. The apparatus of claim 14, further comprising a current variable value map of assert statements saver coupled to said statement traverser.

16. The apparatus of claim 14, further comprising:
- an assert statement determiner coupled to said statement traverser and to said assert chain creator;
  - a variable use mapping adder coupled to said assert statement determiner; and
  - an assert statement reference mapping deleter coupled to said assert statement determiner.
17. The apparatus of claim 14, further comprising a current variable value map of assert statements restorer coupled to said statement traverser.
18. An apparatus for generating factored assert chains for a program in a compiler, the apparatus comprising:
- an assert statement generator;
  - a dominator tree creator coupled to said assert statement generator;
  - an initialized map of assert statements creator coupled to said assert statement generator;
- and
- an assert chain search procedure caller coupled to said initialized map of assert statements creator and to said dominator tree creator.
19. The apparatus of claim 18, wherein said assert statement generator includes:
- an if...then...else statement finder; and
  - an assert statement inserter coupled to said if...then...else statement finder.
20. The apparatus of claim 18, wherein said assert statement generator includes:

a do loop finder; and

an assert statement inserter coupled to said do loop finder.

21. The apparatus of claim 18, wherein said assert chain search procedure caller includes:

a statement traverser;

a variable use finder coupled to said statement traverser;

a variable use traverser coupled to said statement traverser and said variable use finder;

an available assert statement determiner coupled to said variable user traverser;

an assert chain creator coupled to said available assert statement determiner and to said variable use traverser; and

an assert chain search procedure iterative caller coupled to said statement traverser.

22. The apparatus of claim 21, wherein said assert chain search procedure caller further comprises a current variable value map of assert statements saver coupled to said statement traverser.

23. The apparatus of claim 21, wherein said assert chain search procedure caller further comprises:

an assert statement determiner coupled to said statement traverser and to said assert chain creator;

a variable use mapping adder coupled to said assert statement determiner; and

an assert statement reference mapping deleter coupled to said assert statement determiner.

24. The apparatus of claim 21, wherein said assert chain search procedure caller further comprises a current variable value map of assert statements restorer coupled to said statement traverser.

25. An apparatus for generating factored assert chains from assert statements in a program using a compiler, the program having a dominator tree, wherein the method has as input parameter *X* initially assigned an entry node of said dominator tree, the apparatus comprising:

for each statement *S* in *X*:

means for finding each variable use in the statement;

for each variable use in the statement:

means for determining if there is an available assert statement which defines information about the corresponding variable for the variable use;

means for making an assert chain from the variable use to the available assert statement if there is an available assert statement which defines information about the corresponding variable for the variable use; and

means for iteratively calling the method using each child of *X* in the dominator tree as a parameter.

26. The apparatus of claim 25, further comprising means for saving a current value in a map of assert statements for each variable.

27. The apparatus of claim 25, further comprising:

means for determining if the statement is an assert statement after said making;

means for adding a mapping from each variable use in the statement to an assert statement if the statement is an assert statement; and

means for deleting a mapping of any reference to an assert statement for each definition in the statement if the statement is not an assert statement.

28. The apparatus of claim 25, further comprising means for restoring a current value of a map of assert statements for each variable after said iteratively calling.

29. The apparatus of claim 25, wherein an assert statement is a statement identifying known information regarding a variable.

30. An apparatus for generating factored assert chains for a program in a compiler, the apparatus comprising:

means for generating one or more assert statements in a basic block of the program;

means for creating a dominator tree for said basic block;

means for creating an initialized map of assert statements for each variable in said basic block; and

means for calling an assert chain search procedure with entry as a parameter, wherein entry is a root node of said dominator tree.

31. The apparatus of claim 30, wherein said means for generating includes:

means for finding an if...then...else statement in said basic block, said if...then...else statement having a condition, a then portion, and an else portion;



means for inserting an assert statement indicating that said condition is true in said then portion; and

means for inserting an assert statement indicating that said condition is false in said else portion.

32. The apparatus of claim 30, wherein said means for generating includes:

means for finding a do loop, said do loop having a non-constant stride; and

means for inserting an assert statement indicating that said stride is not equal to zero.

33. The apparatus of claim 30, wherein said assert chain search procedure has as input parameter *X*, the assert chain search procedure comprising:

for each statement *S* in *X*:

means for finding each variable use in the statement;

for each variable use in the statement:

means for determining if there is an available assert statement which defines information about the corresponding variable for the variable use;

means for making an assert chain from the variable use to the available assert statement if there is an available assert statement which defines information about the corresponding variable for the variable use; and

means for iteratively calling the method using each child of *X* in the dominator tree as a parameter.

34. The apparatus of claim 33, wherein said assert chain search procedure further comprises means for saving a current value in a map of assert statements for each variable.

35. The apparatus of claim 33, wherein said assert chain search procedure further comprises:  
means for determining if the statement is an assert statement after said making;  
means for adding a mapping from each variable use in the statement to an assert statement if the statement is an assert statement; and

means for deleting a mapping of any reference to an assert statement for each definition in the statement if the statement is not an assert statement.

36. The apparatus of claim 33, wherein said assert chain search procedure further comprises means for restoring a current value of a map of assert statements for each variable after said iteratively calling.

37. The apparatus of claim 33, wherein an assert statement is a statement identifying known information regarding a variable.

38. A program storage device readable by a machine, tangibly embodying a program of instructions executable by the machine to perform a method for generating factored assert chains from assert statements in a program using a compiler, the program having a dominator tree, wherein the method has as input parameter *X* initially assigned an entry node of said dominator tree, the method comprising:

for each statement *S* in *X*:

finding each variable use in the statement;

for each variable use in the statement:

    determining if there is an available assert statement which defines

information about the corresponding variable for the variable use;

    making an assert chain from the variable use to the available assert

statement if there is an available assert statement which defines information about

the corresponding variable for the variable use; and

iteratively calling the method using each child of  $X$  in the dominator tree as a parameter.

39. A program storage device readable by a machine, tangibly embodying a program of instructions executable by the machine to perform a method for generating factored assert chains for a program in a compiler, the method comprising:

    generating one or more assert statements in a basic block of the program;

    creating a dominator tree for said basic block;

    creating an initialized map of assert statements for each variable in said basic block; and

    calling an assert chain search procedure with entry as a parameter, wherein entry is a root node of said dominator tree.